UNITED STATES UTILITY PATENT APPLICATION

FOR

A Method, System, and Apparatus for

Memory Compression with flexible in-memory Cache

Inventors:

Siva Ramakrishnan

Prepared by:
Michael Nesheiwat
Patent Attorney

intel®

Intel Corporation
2111 N.E. 25th Avenue;
JF3-147
Hillsboro, OR 97124
Phone: (503) 712-8918
Facsimile: (503) 264-1729

EV325527887US

BACKGROUND

1. Field

The present disclosure pertains to the field of memory compression. More particularly, the present disclosure pertains to memory compression utilizing an internal cache residing in main memory.

[0001] 2. Description of Related Art

[0002] Memory compression is utilized for reducing large memory requirements, such as, an enterprise server application by compressing data before storing it into memory. Consequently, a reduction in memory costs, power requirements, and server size is achieved.

[0003] Some applications using compressed memory data require different amounts of uncompressed data amounts of cache to alleviate latency impacts. However, typical compression architectures are not flexible for accommodating different cache memory sizes required for different applications.

[0004] Typically, memory compression may be achieved by utilizing a separate external Dynamic Random Access Memory (DRAM) for storing frequently accessed uncompressed data for alleviating the impact of decompression latency. For example, the DRAM may be placed outside the memory interface through a separate memory address/data path in order to have a large cache. However, this incurs the extra cost for both the pins for connecting to the external cache and the cost of the external DRAM. Furthermore, an increase in design and validation costs arises because of the need to test and validate the external cache and the additional interface and an increase in material

costs due to an increase in board size and power requirements.

[0005] Another typical solution is embedded DRAM (eDRAM). However, the current eDRAM solutions (4 and 8 MB) are insufficient to handle server applications that utilize at least 32 MB of memory. In addition eDRAM cache increases the cost of the platform.

Brief Description of the Figures

[0006] The present invention is illustrated by way of example and not limitation in the Figures of the accompanying drawings.

[0007] Figure 1 illustrates an apparatus utilized in accordance with an embodiment

[0008] Figure 2 illustrates a method utilized in accordance with an embodiment.

[0009] Figure 3 illustrates a system in accordance with one embodiment.

[0010] Figure 4 illustrates a system in accordance with one embodiment.

Detailed Description

[0011] The following description provides method, system and apparatus for a flexible compression architecture utilizing internal cache residing in mainmemory. In the following description, numerous specific details are set forth in order to provide a more thorough understanding of the present invention. It will be appreciated, however, by one skilled in the art that the invention may be practiced without such specific details. Those of ordinary skill in the art, with the included descriptions, will be able to implement appropriate logic circuits without undue experimentation.

[0012] As previously described, various problem exist for typical memory compression architectures. In contrast, in one aspect, the claimed subject matter utilizes a main memory for storing compression cache data. In another aspect, the claimed subject matter depicts a flexible compression architecture that may enable expansion of the compression cache by facilitating tag expansion. In yet another aspect, the claimed subject matter depicts a victim buffer and prioritization scheme for alleviating performance impactsed by of compression and decompression operations.

[0013] Figure 1 illustrates an apparatus utilized in accordance with an embodiment. In one aspect and embodiment, the apparatus depicts a novel and flexible memory compression architecture that enables expansion of a compression cache by facilitating tag expansion. In yet another aspect, the apparatus depicts a victim buffer and prioritization scheme for alleviating performance impacts association of with

compression and decompression operations. Furthermore, the apparatus depicts utilizing main memory for storing compression cache data.

[0014] The main memory 104 is coupled to a memory interface 102 via a memory controller 112. .In one aspect, the main memory 104 stores compression cache data and comprises a compression cache 110, compressed memory 108, and a compressed memory pointer table (CMPT) 106. The compression cache 110 may store the uncompressed data and may be organized as a sectored cache, wherein the associated tags are on-die. For example, on-die refers to the tags being incorporated within an integrated device, such as, a processor or cache memory or any integrated device that utilizes the claimed subject matter. In contrast, the compressed memory 108 may store the compressed data and the CMPT may store the pointers to the compressed data for possible cache block addresses. In one embodiment, the CMPT stores the pointers to the compressed data for all possible cache block addresses.

[0015] In one embodiment, the apparatus assigns a higher priority to compressed memory read operations in comparison to other operations, such as, write accesses to compressed memory and other read operations.

[0016] The memory interface 102 comprises the compression cache tag 114, the victim buffer 116, CMPT cache 122 and offset calculator 124, and the compression engine and decompression engine 120 and 128, respectively. In one embodiment, the victim buffer stores the entries that have been evicted from the compression cache.

[0017] In one embodiment, if data from the compression cache needs to be vacated to store another uncompressed data in its place, the least recently used entry can be vacated. However, the claimed subject matter is not limited to least recently used. For

example, other eviction techniques, such as random or round robin may be implemented. In one embodiment, evictions for entries in the victim buffer are First In First Out.

[0018] In one embodiment, the CMPT cache stores the most recently accessed CMPT entries. A more detailed description of the apparatus will be discussed in the following paragraphs and pages.

[0019] In one embodiment, the apparatus 100 is utilized for compression and decompression functions. For example, the compression cache and victim buffer receive incoming memory addresses wherein a tag match operation is performed. If there is a compression cache hit, a read/write hit signal is enabled and forwarded to the memory controller for scheduling an uncompressed data access in the compression cache 110. Otherwise, in case of compression cache miss, if the access is a hit in the victim buffer, the data is directly supplied to the requester. A more detailed description of a compression cache hit is discussed in paragraph 22.

[0020] Otherwise, in the event of a read miss, the pointer to the compressed memory location is obtained either from the CMPT cache 122 or from the CMPT 106 in main memory 104. . The CMPT stores the pointer (an address) to the compressed data that is being requested. In one embodiment, it will take one access to get this pointer and then another access to get the actual compressed data, a small cache in the memory interface is used to store the most recently used compressed data pointers. In one embodiment, the CMPT cache is first searched for the pointer. If this cache does not have the pointer, then the pointer is obtained from the main memory itself first. Then the location pointed to by the pointer is accessed subsequently to obtain the actual compressed memory data. The

[0021] Subsequently, After the pointer is obtained, . Consequently, the compressed

memory location designated by the pointer is accessed and the data is forwarded to the decompression engine 128. Subsequently, the decompressed data is output from the decompression engine 128 and is forwarded to the requester of the initial memory access of the incoming address. Likewise, in one embodiment the decompressed data is subsequently written to the compression engine cache in order to store the most recently accessed memory item in uncompressed form Before doing this, a victim data from the compression cache is chosen and vacated to the victim buffer.. In the event of a compression cache write miss, the data is compressed by the compression engine and is stored in the compressed memory location based at least in part on a pointer that may be indicated by a CMPT cache entry. Otherwise, if the pointer is not available in the CMPT cache, then a corresponding CMP table entry in the main memory 104 is accessed by using a CMPT offset calculator 124.

[0022] The CMPT (table) stores pointers to compressed data sequentially based on memory address for which the data is compressed. These pointers are of fixed size. In one embodiment, the CMPT offset calculator provides the offset relative to the start of the table based on the actual address of the data being compressed. For example, it may be used in conjuction with the fixed starting address of the table to locate the pointer.

[0023] As previously discussed in paragraph 20 for the condition of a compression cache hit, a read/write hit signal is enabled and forwarded to the memory controller for scheduling an uncompressed data access in the compression cache 110. In one embodiment, the data is forwarded to the requester of the initial memory access of the incoming address if the data resides in the victim buffer. Alternatively, the data is forwarded to the requester of the initial memory access of the incoming address if the

data resides in the compression cache since the data is uncompressed.

[0024] In one embodiment, the latency of compression cache data accesses is identical to an uncompressed regular memory access. In one aspect, the victim buffer may be utilized to alleviate the impact of increased bandwidth demand due to additional memory traffic by storing evicted lines from the compression cache. As previously described, the apparatus assigns a higher priority to compressed memory read operations in comparison to other operations, such as, write accesses to compressed memory and other read operations.

[0025] Figure 2 depicts a flowchart for a method in accordance with one embodiment. In one aspect, the method depicts a flexible scheme for access to compressed and uncompressed memory. Upon receiving a request for a memory access, a tag match is performed, as depicted by a block 202. In one embodiment, the tag match is performed at a compression cache and a victim buffer.

If the tag match results in a hit, the uncompressed data is retrieved from a compression cache, as depicted by a block 204. Otherwise, for a read miss, the method depicts locating a pointer and subsequently finding a compressed memory location based at least in part on the pointer, as depicted by a block 206. Otherwise, for a write miss, compressing the data by a compression engine and storing it in the compressed memory location based at least in part on a pointer in a CMPT cache entry or based on a CMPT offset calculator, as depicted by a block 208.

[0026] Figure 3 depicts a system in accordance with one embodiment. The system in one embodiment is a processor 302 that is coupled to a chipset 304 that is coupled to a memory 306. For example, the chipset performs and facilitates various operations, such

as, memory transactions between the processor and memory. In one embodiment, the system comprises one or all of the previous embodiments depicted in connection with Figures 1-2 of the specification to allow for a flexibly memory compression architecture. In one embodiment, the memory interface discussed in connection with Figure 1 may be incorporated within the chipset. Alternatively, in another embodiment, the memory interface discussed in connection with Figure 1 may be incorporated within the processor.

[0027] Figure 4 depicts a system in accordance with one embodiment. In this embodiment, a processor 402 is coupled to a memory bridge 404, such as, a chipset. The memory bridge incorporates the same logic blocks as memory interface 102 depicted earlier in connection with Figure 1. Furthermore, the memory bridge is coupled to a main memory, which in this embodiment incorporates the same logic blocks as main memory 104 depicted earlier in connection with Figure 1.

[0028] While certain exemplary embodiments have been described and shown in the accompanying drawings, it is to be understood that such embodiments are merely illustrative of and not restrictive on the broad invention, and that this invention not be limited to the specific constructions and arrangements shown and described, since various other modifications may occur to those ordinarily skilled in the art upon studying this disclosure.